DS6003 Spark Assignment - as3ek

Motivation : To understand the workflow of pyspark using a standard logistic regression problem on the titanic dataset[1].

The objective of the defined problem is to predict if a passenger survived the titanic sinking based on his/her age.

ROC curve is used to explain the model performance and hence the correlation between the age and the passenger surviving.

[1] https://www.kaggle.com/c/titanic/data

Code Snippet

In [14]:
```
# create train/test sets
seed = 42
(testDF, trainingDF) = df.randomSplit((0.20, 0.80), seed=seed)
print ('training set N = {}, test set N = {}'.format(trainingDF.count(),testDF.count()))
```

```
training set N = 143, test set N = 40
```

In [15]:
```
from pyspark.ml.linalg import Vectors, VectorUDT

# make a user defined function (udf)
sqlc.registerFunction("oneElementVec", lambda d: Vectors.dense([d]), returnType=VectorUDT())

# vectorize the data frames
trainingDF = trainingDF.selectExpr("Survived", "oneElementVec(Age) as Age")
testDF = testDF.selectExpr("Survived", "oneElementVec(Age) as Age")

print(trainingDF.orderBy(trainingDF.Age.desc()).limit(5))
```

```
DataFrame[Survived: bigint, Age: vector]
```

In [16]:
```
# Renaming columns for happiness
trainingDF = trainingDF.withColumnRenamed("Survived", "label").withColumnRenamed("Age", "features")
testDF = testDF.withColumnRenamed("Survived", "label").withColumnRenamed("Age", "features")
```

In [17]:
```
from pyspark.ml.classification import *

lr = LogisticRegression()
lrModel = lr.fit(trainingDF)
```

In [18]:
```
type(lrModel)
```

Out[18]:
```
pyspark.ml.classification.LogisticRegressionModel
```

In [19]:
```
predictionsAndLabelsDF = lrModel.transform(testDF)

print(predictionsAndLabelsDF.orderBy(predictionsAndLabelsDF.label.desc()).take(5))
```

```
[Row(label=1, features=DenseVector([4.0]), rawPrediction=DenseVector([-2.0492, 2.0492]), probability=DenseVector
([0.1141, 0.8859]), prediction=1.0), Row(label=1, features=DenseVector([4.0]), rawPrediction=DenseVector([-2.0492,
2.0492]), probability=DenseVector([0.1141, 0.8859]), prediction=1.0), Row(label=1, features=DenseVector([11.0]), ra
wPrediction=DenseVector([-1.7579, 1.7579]), probability=DenseVector([0.147, 0.853]), prediction=1.0), Row(label=1,
features=DenseVector([14.0]), rawPrediction=DenseVector([-1.6331, 1.6331]), probability=DenseVector([0.1634, 0.836
6]), prediction=1.0), Row(label=1, features=DenseVector([17.0]), rawPrediction=DenseVector([-1.5083, 1.5083]), prob
ability=DenseVector([0.1812, 0.8188]), prediction=1.0)]
```

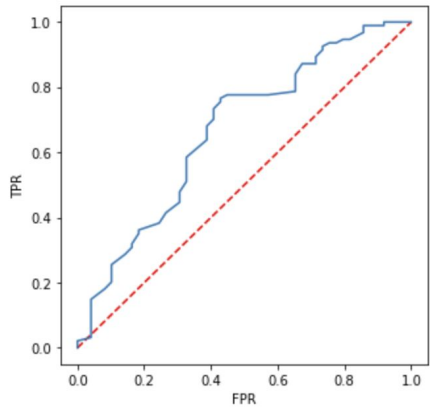
The data was read from the s3 bucket after previously uploading it to the bucket. It was subsequently read into a spark dataframe via parquet.

After vectorising the feature required to predict the identified label, the data was split into train and test set.

A logistic regression model was then trained on this data and predictions were made on the test data, the performance of the model was evaluated using various model statistics and an ROC curve.

Visualisation (ROC Curve)

In [25]:
```
import matplotlib.pyplot as plt
plt.figure(figsize=(5,5))
plt.plot([0, 1], [0, 1], 'r--')
plt.plot(lrModel.summary.roc.select('FPR').collect(),
         lrModel.summary.roc.select('TPR').collect())
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.show()
```



The ROC curve was plotted by extracting the FPR and the TPR from the trained model to assess the performance of the model